**PROJECT REPORT**

Project Name**: EMERGING METHODS FOR EARLY DETECTION OF FOREST FIRES.**

Team id: **PNT2022TMID07051**

Team members: **ANISHA SINGH**

**ARTHI**

**DHARANI**

**LOGAVARSHINI**

**INTRODUCTION**

# Project overview

Wildﬁre, also called forest ﬁre, bush or vegetation ﬁre, can be described as any uncontrolled and non-prescribed combustion or burning of plants in a natural setting such as a forest, grassland, brush land or tundra, which consumes the natural fuels and spreads based on environmental conditions (e.g., wind, topography). Forest ﬁres are a major environmental issue, creating economic and ecological damage while endangering human lives. There are typically about 100,000 wildﬁres in the United States every year. Over 9 million acres of land have been destroyed due to treacherous wildﬁres. It is diﬃcult to predict and detect Forest Fire in a sparsely populated forest area and it is more diﬃcult if the prediction is done using ground- based methods like Camera or Video-Based approach. Satellites can be an important source of data prior and also during the Fire due to its reliability and eﬃciency. The various real-time forest ﬁre detection and prediction approaches, with the goal of informing the local ﬁre authorities.

This is a huge problem which needs to be tackled and thus through this project we provide a way to tackle the issue.

# Purpose

The purpose of the project is to detect the forest ﬁre earlier.

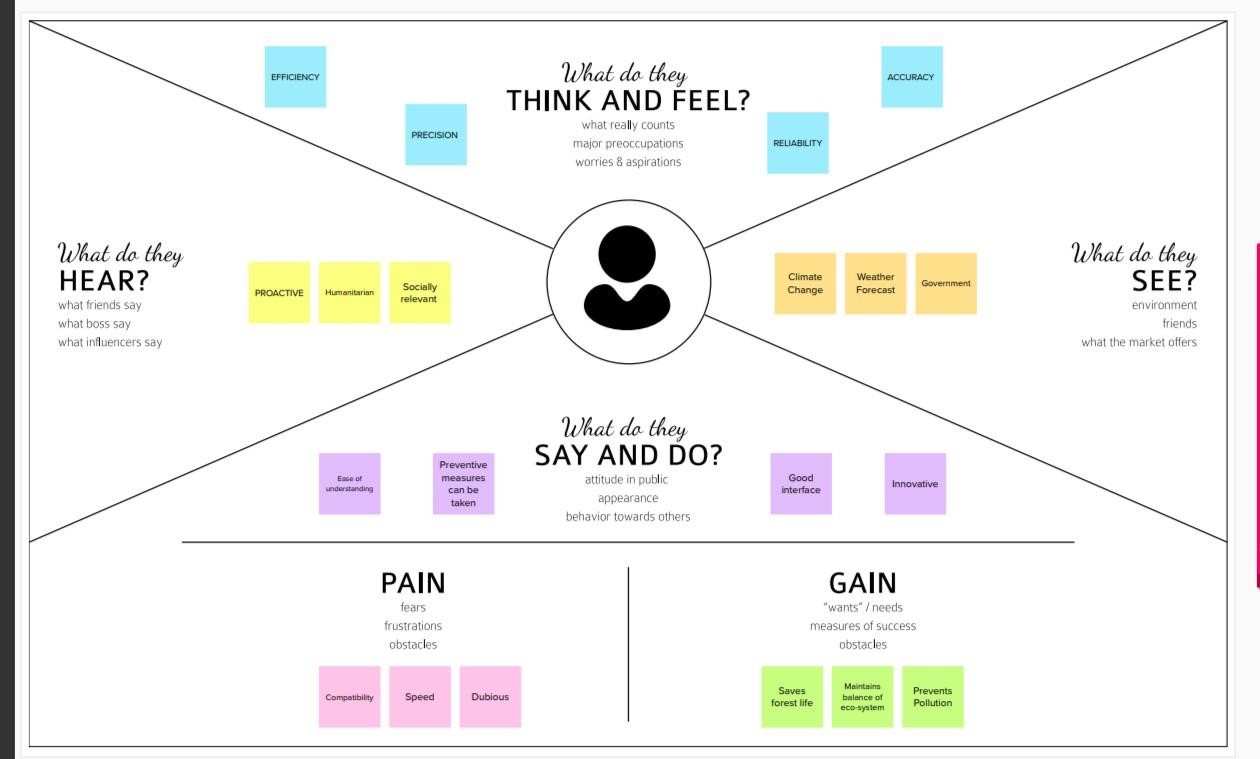
# LITERATURE SURVEY

* 1. **Reference**

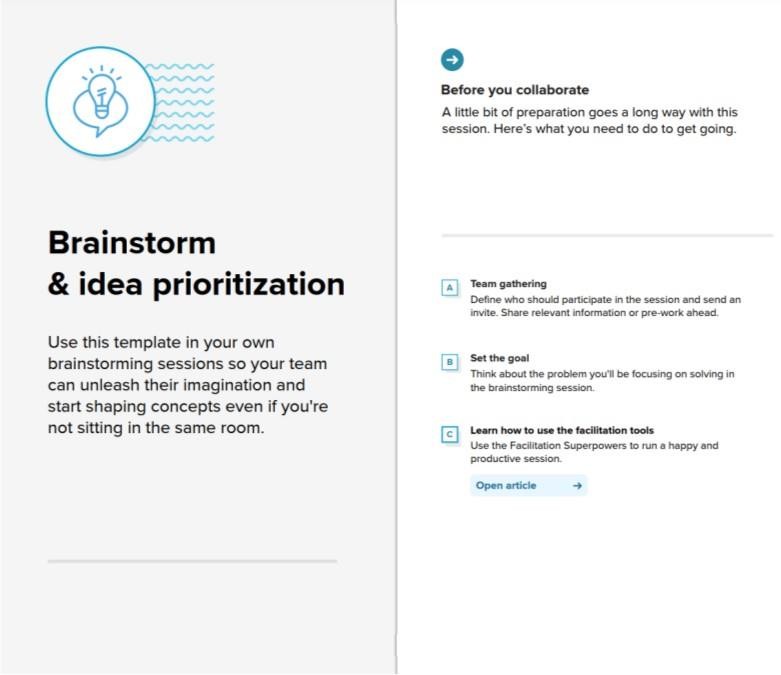
|  |  |  |  |
| --- | --- | --- | --- |
| **S.**  **NO** | **TITL E** | **AUTHO R** | **YEAR** |
| 1. | Image Processing for Forest Fire  Detection. | Priyadharshini | 2016 |
| 2. | Forest ﬁre prediction and detection  system. | Faroudja Abid | 2020 |
| 3. | systematic approaches in managingforest ﬁres . | [AdityaDhall](https://www.sciencedirect.com/science/article/abs/pii/S0143622818311718#!) | 2020 |

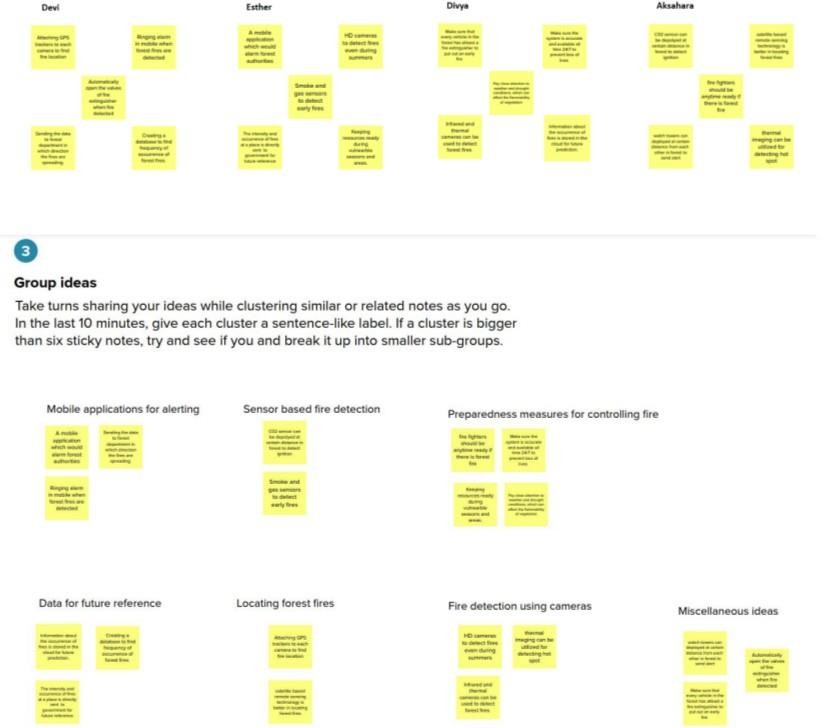
# IDEATION & PROPOSED SOLUTION

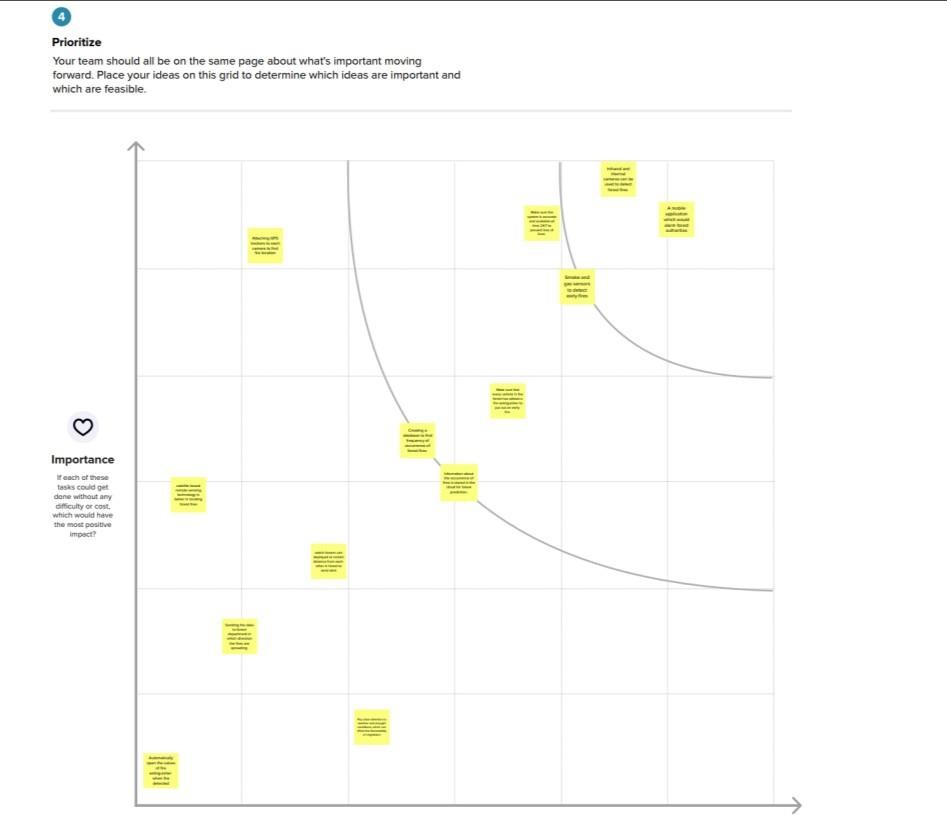
* 1. **Empathy map**



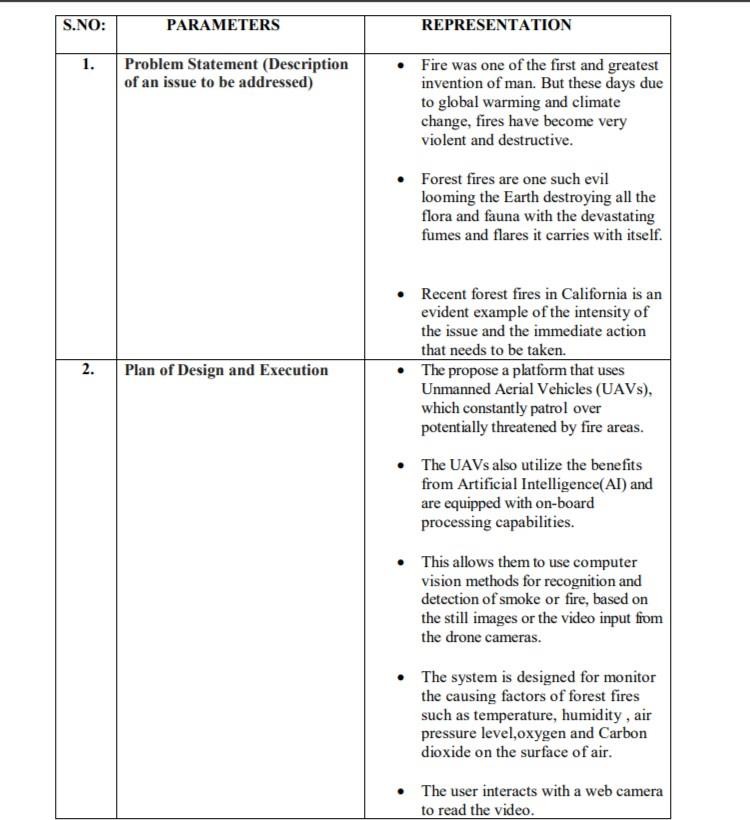
# Ideation & Brainstorming

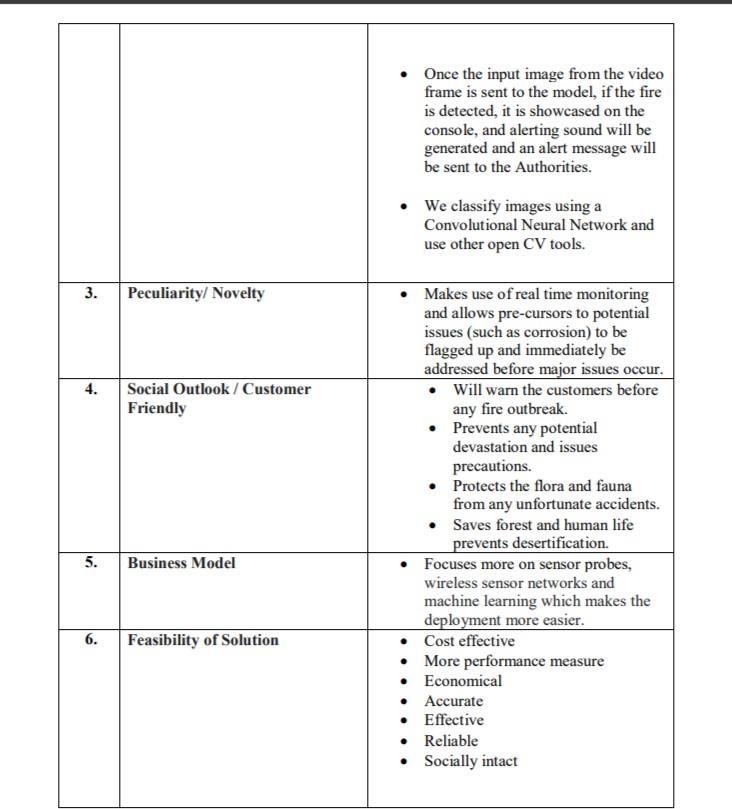




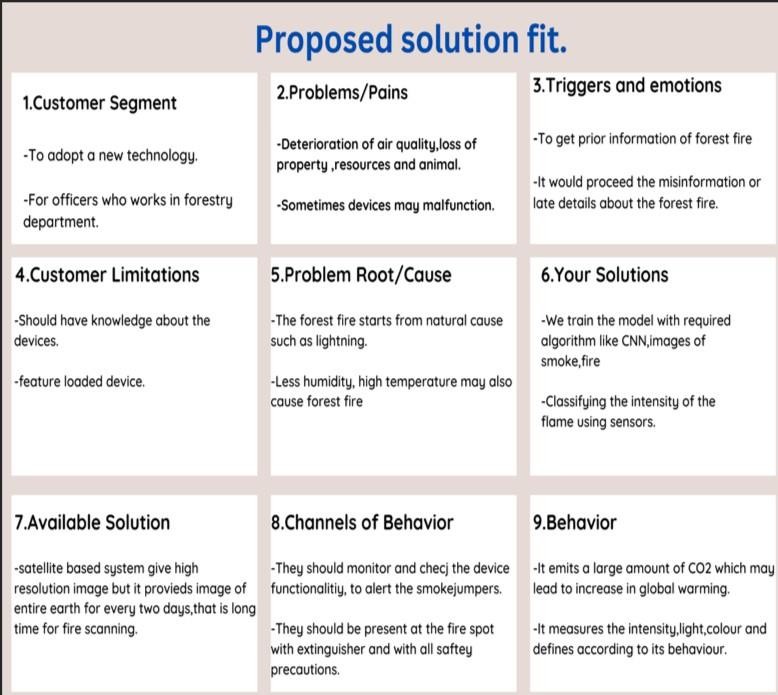


* 1. **Proposed solution**



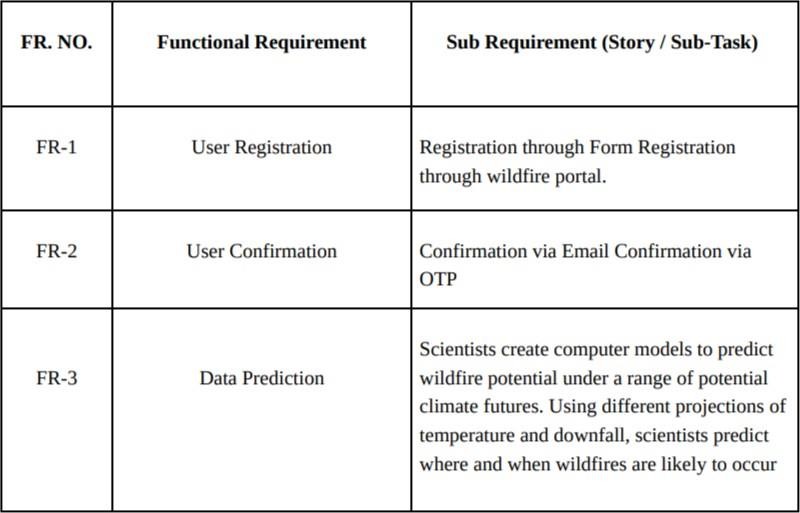


# Problem solution ﬁt

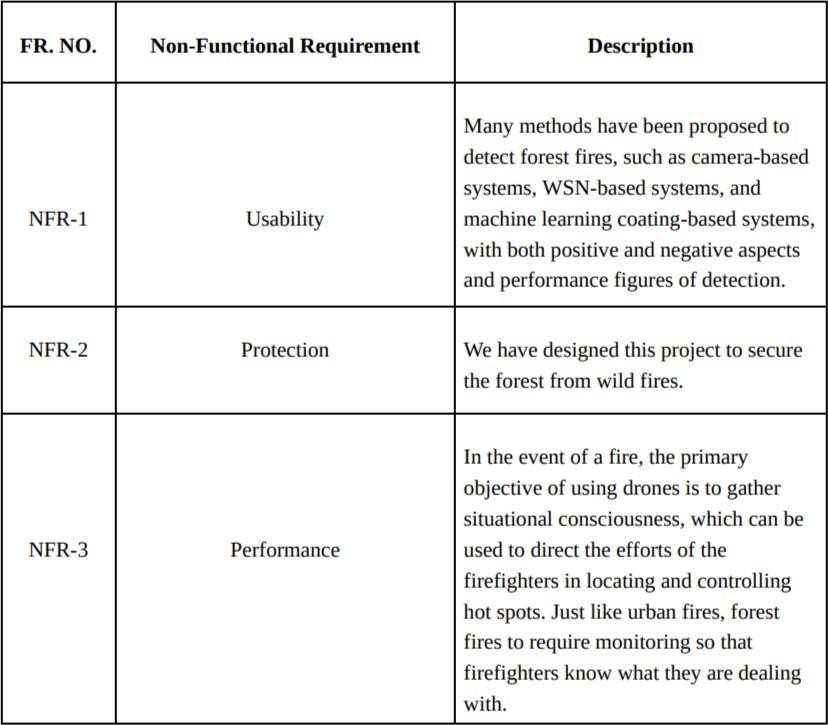


1. **REQUIREMENT ANALYSIS**

# Functional requirement

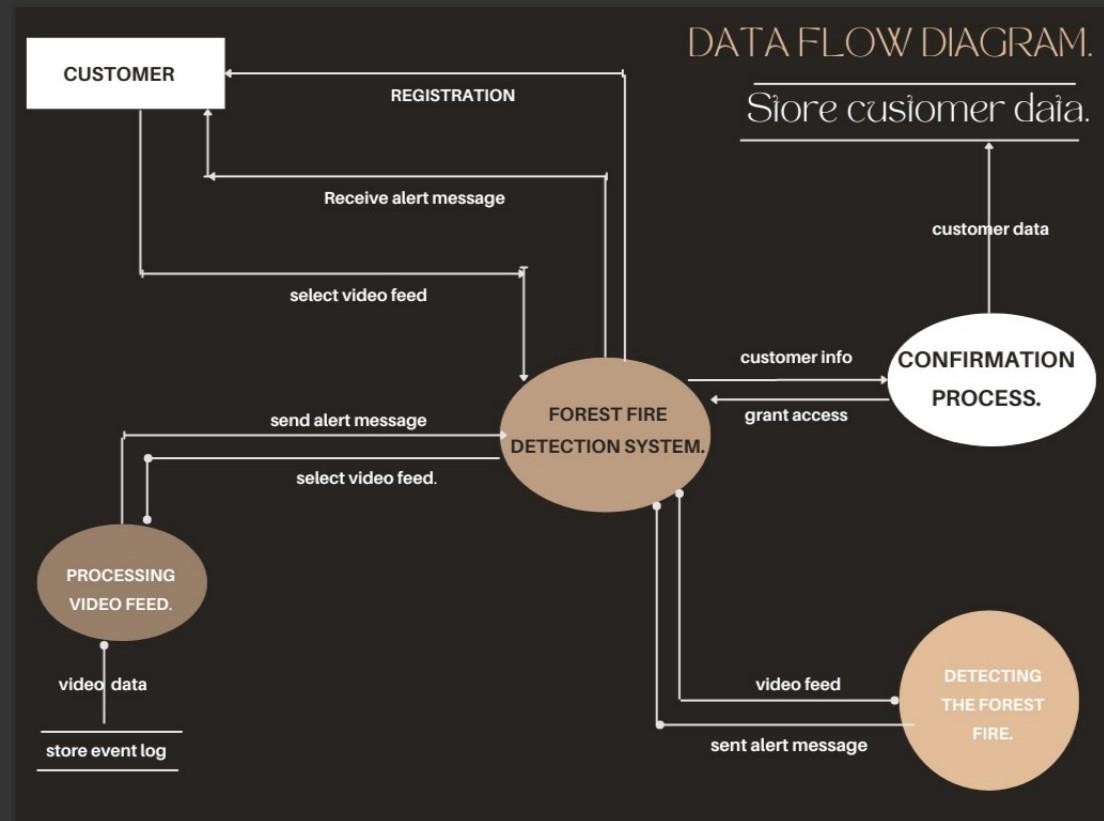


* 1. **Non-Functional requirement**

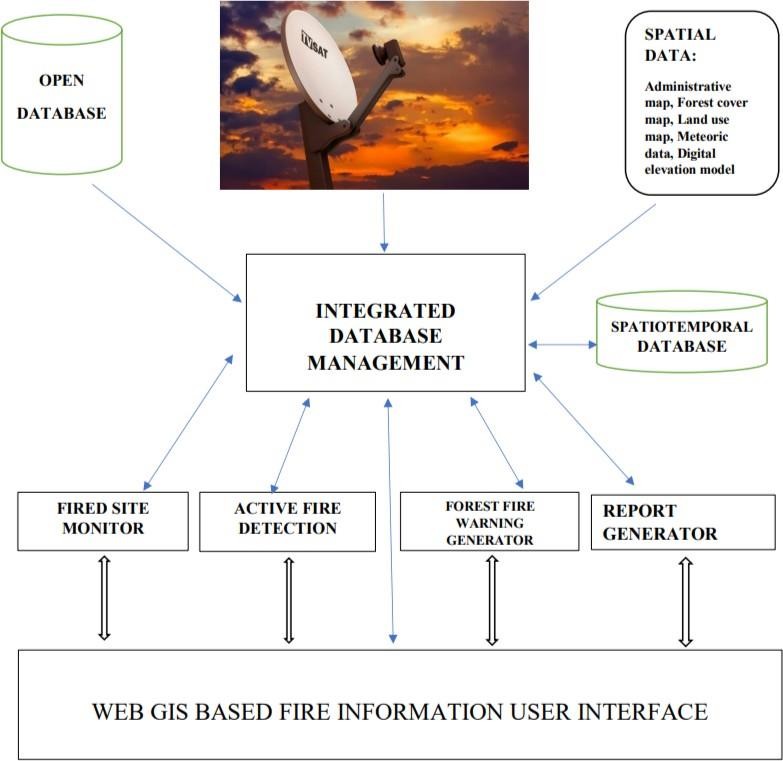


# PROJECT DESIGN

* 1. **Data Flow Diagrams**

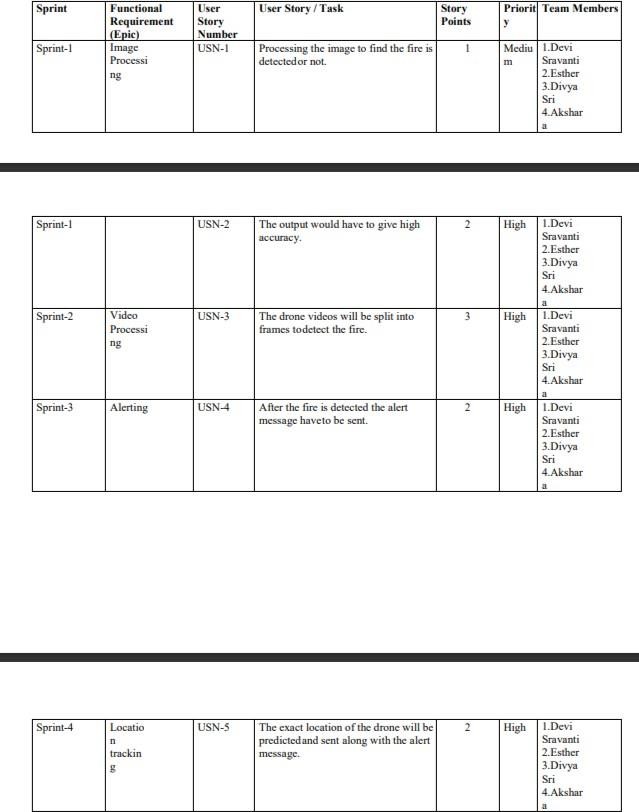


# Solution Architecture

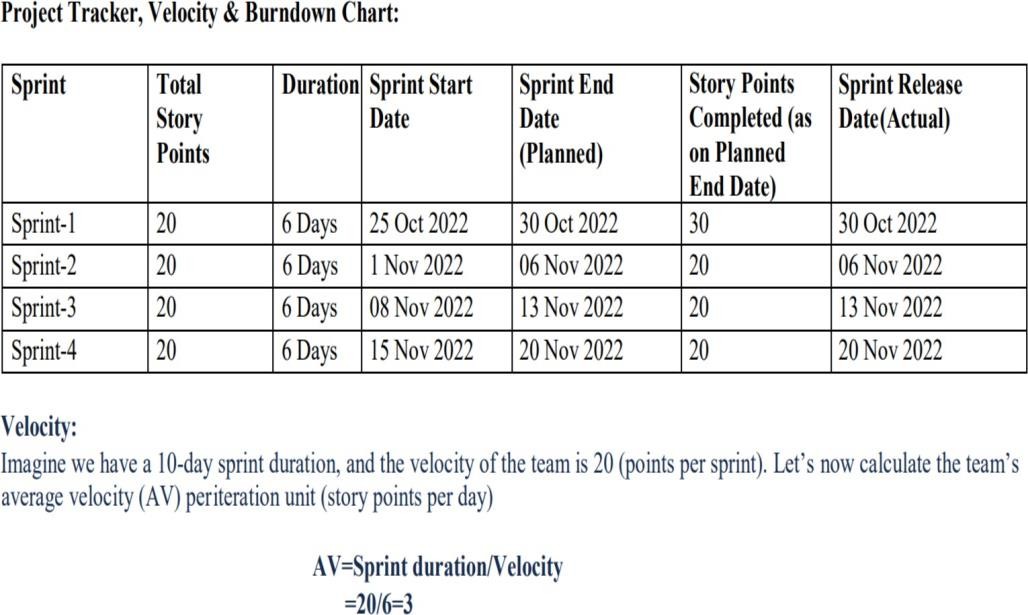


1. **PROJECT PLANNING & SCHEDULING**

# Sprint Planning & Estimation



* 1. **Sprint delivery schedule**



# SPRINT-1 (COLLECTION OF DATATSET)



* 1. **SPRINT-2 (MODEL BUILDING AND CLASSIFICATION) import** tensorflow **as** tf

**import** numpy **as** np

**from** tensorflow **import** keras

**import** os

**import** cv2

**from** tensorflow.keras.preprocessing.image **import** ImageDataGenerator

**from** tensorflow.keras.preprocessing **import** image

**import** matplotlib.pyplot **as** plt

train **=** ImageDataGenerator(rescale**=**1**/**255) test **=** ImageDataGenerator(rescale**=**1**/**255)

train\_dataset **=** train**.**flow\_from\_directory(r"/content/drive/MyDrive/train\_set",

target\_size**=**(150,150), batch\_size **=** 32,

class\_mode **=** 'binary')

test\_dataset **=** test**.**flow\_from\_directory(r"/content/drive/MyDrive/test\_set", target\_size**=**(150,150),

batch\_size **=**32, class\_mode **=** 'binary')

Found 442 images belonging to 2 classes. Found 121 images belonging to 2 classes. test\_dataset**.**class\_indices

{'forest': 0, 'with fire': 1} model **=** keras**.**Sequential()

model**.**add(keras**.**layers**.**Conv2D(32,(3,3),activation**=**'relu',input\_shape**=**(150,150,3))) model**.**add(keras**.**layers**.**MaxPool2D(2,2)) model**.**add(keras**.**layers**.**Conv2D(64,(3,3),activation**=**'relu')) model**.**add(keras**.**layers**.**MaxPool2D(2,2)) model**.**add(keras**.**layers**.**Conv2D(128,(3,3),activation**=**'relu')) model**.**add(keras**.**layers**.**MaxPool2D(2,2)) model**.**add(keras**.**layers**.**Conv2D(128,(3,3),activation**=**'relu')) model**.**add(keras**.**layers**.**MaxPool2D(2,2))

model**.**add(keras**.**layers**.**Flatten()) model**.**add(keras**.**layers**.**Dense(512,activation**=**'relu')) model**.**add(keras**.**layers**.**Dense(1,activation**=**'sigmoid')) model**.**summary()

Model: "sequential"

Layer (type) Output Shape Param #

=================================================================

conv2d (Conv2D) (None, 148, 148, 32) 896

max\_pooling2d (MaxPooling2D (None, 74, 74, 32) 0

)

conv2d\_1 (Conv2D) (None, 72, 72, 64) 18496

max\_pooling2d\_1 (MaxPooling (None, 36, 36, 64) 0 2D)

conv2d\_2 (Conv2D) (None, 34, 34, 128) 73856

max\_pooling2d\_2 (MaxPooling (None, 17, 17, 128) 0 2D)

conv2d\_3 (Conv2D) (None, 15, 15, 128) 147584

max\_pooling2d\_3 (MaxPooling (None, 7, 7, 128) 0

2D)

|  |  |  |
| --- | --- | --- |
| flatten (Flatten) | (None, 6272) | 0 |
| dense (Dense) | (None, 512) | 3211776 |
| dense\_1 (Dense) | (None, 1) | 513 |

=================================================================

Total params: 3,453,121

Trainable params: 3,453,121

Non-trainable params: 0

model**.**compile(optimizer**=**'adam',loss**=**'binary\_crossentropy',metrics**=**['accuracy']) r **=** model**.**fit(train\_dataset,

epochs **=** 10,

validation\_data **=** test\_dataset) Epoch 1/10

14/14 [==============================] - 156s 11s/step - loss: 0.5977 - accuracy: 0.7353 -

val\_loss: 0.2603 - val\_accuracy: 0.9256 Epoch 2/10

14/14 [==============================] - 31s 2s/step - loss: 0.3629 - accuracy: 0.8507 - val

\_loss: 0.1304 - val\_accuracy: 0.9752

Epoch 3/10

14/14 [==============================] - 29s 2s/step - loss: 0.2061 - accuracy: 0.9276 - val

\_loss: 0.0353 - val\_accuracy: 0.9917 Epoch 4/10

14/14 [==============================] - 30s 2s/step - loss: 0.1594 - accuracy: 0.9457 - val

\_loss: 0.0253 - val\_accuracy: 1.0000 Epoch 5/10

14/14 [==============================] - 29s 2s/step - loss: 0.1493 - accuracy: 0.9434 - val

\_loss: 0.0274 - val\_accuracy: 1.0000 Epoch 6/10

14/14 [==============================] - 30s 2s/step - loss: 0.1211 - accuracy: 0.9548 - val

\_loss: 0.0222 - val\_accuracy: 1.0000 Epoch 7/10

14/14 [==============================] - 30s 2s/step - loss: 0.1200 - accuracy: 0.9525 - val

\_loss: 0.1301 - val\_accuracy: 0.9256 Epoch 8/10

14/14 [==============================] - 31s 2s/step - loss: 0.1361 - accuracy: 0.9434 - val

\_loss: 0.0206 - val\_accuracy: 0.9917 Epoch 9/10

14/14 [==============================] - 32s 2s/step - loss: 0.1498 - accuracy: 0.9412 - val

\_loss: 0.0352 - val\_accuracy: 1.0000 Epoch 10/10

14/14 [==============================] - 32s 2s/step - loss: 0.0850 - accuracy: 0.9661 - val

\_loss: 0.0065 - val\_accuracy: 1.0000 model**.**save("forest1.h5")

predictions **=** model**.**predict(test\_dataset) predictions **=** np**.**round(predictions)

4/4 [==============================] - 6s 1s/step

predictions array([[1.],

[1.],

[0.],

[1.],

[0.],

[1.],

[0.],

[0.],

[1.],

[1.],

[0.],

[0.],

[0.],

[1.],

[0.],

[0.],

[0.],

[1.],

[0.],

[0.],

[1.],

[0.],

[0.],

[0.],

[0.],

[0.],

[0.],

[0.],

[0.],

[1.],

[0.],

[0.],

[0.],

[0.],

[0.],

[0.],

[0.],

[1.],

[1.],

[0.],

[0.],

[1.],

[1.],

[0.],

[1.],

[1.],

[0.],

[1.],

[1.],

[1.],

[0.],

[0.],

[0.],

[1.],

[0.],

[1.],

[0.],

[0.],

[1.],

[0.],

[0.],

[1.],

[0.],

[0.],

[0.],

[1.],

[0.],

[1.],

[0.],

[0.],

[0.],

[1.],

[1.],

[0.],

[1.],

[1.],

[0.],

[1.],

[0.],

[1.],

[1.],

[0.],

[0.],

[1.],

[1.],

[1.],

[0.],

[0.],

[0.],

[1.],

[0.],

[1.],

[0.],

[1.],

[0.],

[1.],

[1.],

[1.],

[1.],

[0.],

[0.],

[0.],

[0.],

[0.],

[1.],

[0.],

[0.],

[0.],

[1.],

[1.],

[1.],

[0.],

[0.],

[1.],

[0.],

[0.],

[1.],

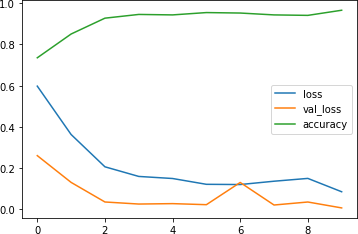
[0.],

[0.],

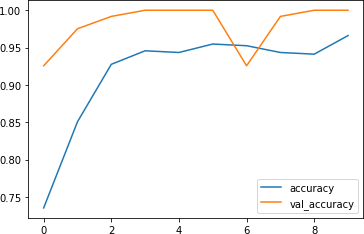
[0.],

[1.]], dtype=float32) print(len(predictions)) 121

**import** matplotlib.pyplot **as** plt plt**.**plot(r**.**history['loss'], label**=**'loss') plt**.**plot(r**.**history['val\_loss'], label**=**'val\_loss') plt**.**plot(r**.**history['accuracy'], label**=**'accuracy') plt**.**legend()



plt**.**plot(r**.**history['accuracy'], label**=**'accuracy') plt**.**plot(r**.**history['val\_accuracy'], label**=**'val\_accuracy') plt**.**legend()



**def** predictImage(filename):

img1 **=** image**.**load\_img(filename,target\_size**=**(150,150)) plt**.**imshow(img1)

Y **=** image**.**img\_to\_array(img1) X **=** np**.**expand\_dims(Y,axis**=**0) val **=** model**.**predict(X) print(val)

**if** val **==** 1: plt**.**xlabel("Fire")

**elif** val **==** 0: plt**.**xlabel("No Fire")

predictImage(r"/content/drive/MyDrive/test\_set/with fire/19464620\_401.jpg") 1/1 [==============================] - 0s 147ms/step

[[1.]]



predictImage(r"/content/drive/MyDrive/test\_set/forest/091318\_LH\_forest\_loss\_main\_FREE.jpg") 1/1 [==============================] - 0s 32ms/step

[[0.]]



predictImage(r"/content/drive/MyDrive/train\_set/with fire/with fire (100).jpg") [[1.]]

1/1 [==============================] - 0s 31ms/step

[[1.]]

[[1.0]]



predictImage(r"/content/drive/MyDrive/test\_set/forest/cold\_daylight\_environment\_1423600\_640x4 27.jpg")

1/1 [==============================] - 0s 31ms/step [[0.]]



predictImage(r"/content/drive/MyDrive/test\_set/with fire/Fire\_2\_696x392.jpg") 1/1 [==============================] - 0s 28ms/step

[[1.]]



predictImage(r"/content/drive/MyDrive/train\_set/forest/with\_fire (104).jpg") 1/1 [==============================] - 0s 80ms/step

[[0.]]



# SPRINT 3:

{

"nbformat": 4,

"nbformat\_minor": 0, "metadata": {

"colab": {

"provenance": []

},

"kernelspec": { "name": "python3",

"display\_name": "Python 3"

},

"language\_info":

{"name": "python"

},

"cells": [

{

"cell\_type": "code", "execution\_count": 1, "metadata": {

"colab": {

"base\_uri": "https:/ localhost:8080/", "height": 35

},

"id": "cm0cXpbvyyBp",

"outputId": "4bffc3ff-b763-4d6d-c12b-02b62f8c32fa"

},

"outputs": [

{

"output\_type": "execute\_result", "data": {

"text/plain":

[

"'/content'"

],

"application/vnd.google.colaboratory.intrinsic+json": { "type": "string"

}

},

"metadata": {}, "execution\_count": 1

}

],

"source":

["pwd"

]

},

{

"cell\_type": "code", "source": [

"!pip install keras\n",

"!pip install tensorﬂow\n", "!pip install opencv-python"

],

"metadata": {

"colab": {

"base\_uri": "https:/ localhost:8080/"

},

"id": "UnpPHFm0y4lm",

"outputId": "543ceb28-d9f5-4c1c-9934-02075c827323"

},

"execution\_count": 2, "outputs": [

{

"output\_type": "stream", "name": "stdout", "text": [

"Looking in indexes: https:/ pypi.org/simple, https:/ us-python.pkg.dev/colab- wheels/public/simple/\n",

"Requirement already satisﬁed: keras in /usr/local/lib/python3.7/dist-packages (2.9.0)\n", "Looking in indexes: https:/ pypi.org/simple, https:/ us-python.pkg.dev/colab-

wheels/public/simple/\n",

"Requirement already satisﬁed: tensorﬂow in /usr/local/lib/python3.7/dist-packages (2.9.2)\n",

"Requirement already satisﬁed: six>=1.12.0 in /usr/local/lib/python3.7/dist-packages (from tensorﬂow) (1.15.0)\n",

"Requirement already satisﬁed: keras<2.10.0,>=2.9.0rc0 in /usr/local/lib/python3.7/dist- packages (from tensorﬂow) (2.9.0)\n",

"Requirement already satisﬁed: libclang>=13.0.0 in /usr/local/lib/python3.7/dist- packages (from tensorﬂow) (14.0.6)\n",

"Requirement already satisﬁed: gast<=0.4.0,>=0.2.1 in /usr/local/lib/python3.7/dist- packages (from tensorﬂow) (0.4.0)\n",

"Requirement already satisﬁed: protobuf<3.20,>=3.9.2 in /usr/local/lib/python3.7/dist- packages (from tensorﬂow) (3.19.6)\n",

"Requirement already satisﬁed: ﬂatbuffers<2,>=1.12 in /usr/local/lib/python3.7/dist- packages (from tensorﬂow) (1.12)\n",

"Requirement already satisﬁed: wrapt>=1.11.0 in /usr/local/lib/python3.7/dist-packages (from tensorﬂow) (1.14.1)\n",

"Requirement already satisﬁed: packaging in /usr/local/lib/python3.7/dist-packages (from tensorﬂow) (21.3)\n",

"Requirement already satisﬁed: tensorﬂow-estimator<2.10.0,>=2.9.0rc0 in

/usr/local/lib/python3.7/dist-packages (from tensorﬂow) (2.9.0)\n",

"Requirement already satisﬁed: h5py>=2.9.0 in /usr/local/lib/python3.7/dist-packages (from

tensorﬂow) (3.1.0)\n",

"Requirement already satisﬁed: absl-py>=1.0.0 in /usr/local/lib/python3.7/dist-packages (from tensorﬂow) (1.3.0)\n",

"Requirement already satisﬁed: astunparse>=1.6.0 in /usr/local/lib/python3.7/dist

packages (from tensorﬂow) (1.6.3)\n",

"Requirement already satisﬁed: typing-extensions>=3.6.6 in /usr/local/lib/python3.7/dist- packages (from tensorﬂow) (4.1.1)\n",

"Requirement already satisﬁed: termcolor>=1.1.0 in /usr/local/lib/python3.7/dist- packages (from tensorﬂow) (2.1.0)\n",

"Requirement already satisﬁed: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.7/dist- packages (from tensorﬂow) (1.50.0)\n",

"Requirement already satisﬁed: google-pasta>=0.1.1 in /usr/local/lib/python3.7/dist- packages (from tensorﬂow) (0.2.0)\n",

"Requirement already satisﬁed: opt-einsum>=2.3.2 in /usr/local/lib/python3.7/dist- packages (from tensorﬂow) (3.3.0)\n",

"Requirement already satisﬁed: keras-preprocessing>=1.1.1 in

/usr/local/lib/python3.7/dist-packages (from tensorﬂow) (1.1.2)\n",

"Requirement already satisﬁed: numpy>=1.20 in /usr/local/lib/python3.7/dist-packages (from tensorﬂow) (1.21.6)\n",

"Requirement already satisﬁed: setuptools in /usr/local/lib/python3.7/dist-packages (from tensorﬂow) (57.4.0)\n",

"Requirement already satisﬁed: tensorboard<2.10,>=2.9 in /usr/local/lib/python3.7/dist-packages (from tensorﬂow) (2.9.1)\n",

"Requirement already satisﬁed: tensorﬂow-io-gcs-ﬁlesystem>=0.23.1 in

/usr/local/lib/python3.7/dist-packages (from tensorﬂow) (0.27.0)\n",

"Requirement already satisﬁed: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.7/dist- packages (from astunparse>=1.6.0->tensorﬂow) (0.38.3)\n",

"Requirement already satisﬁed: cached-property in /usr/local/lib/python3.7/dist- packages (from h5py>=2.9.0->tensorﬂow) (1.5.2)\n",

"Requirement already satisﬁed: werkzeug>=1.0.1 in /usr/local/lib/python3.7/dist- packages (from tensorboard<2.10,>=2.9->tensorﬂow) (1.0.1)\n",

"Requirement already satisﬁed: markdown>=2.6.8 in /usr/local/lib/python3.7/dist- packages (from tensorboard<2.10,>=2.9->tensorﬂow) (3.4.1)\n",

"Requirement already satisﬁed: requests<3,>=2.21.0 in /usr/local/lib/python3.7/dist- packages (from tensorboard<2.10,>=2.9->tensorﬂow) (2.23.0)\n",

"Requirement already satisﬁed: google-auth<3,>=1.6.3 in /usr/local/lib/python3.7/dist- packages (from tensorboard<2.10,>=2.9->tensorﬂow) (2.14.1)\n",

"Requirement already satisﬁed: tensorboard-plugin-wit>=1.6.0 in

/usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorﬂow) (1.8.1)\n", "Requirement already satisﬁed: google-auth-oauthlib<0.5,>=0.4.1 in

/usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorﬂow) (0.4.6)\n", "Requirement already satisﬁed: tensorboard-data-server<0.7.0,>=0.6.0 in

/usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorﬂow) (0.6.1)\n", "Requirement already satisﬁed: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.7/dist-

packages (from google- <2.10,>=2.9>tensorﬂow) (5.2.0)\n",

“Requirement already satisﬁed: rsa<5,>=3.1.4 in /usr/local/lib/python3.7/dist-packages(from google-auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorﬂow) (4.9)\n",

"Requirement already satisﬁed: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.7/dist- packages (from google-auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorﬂow) (0.2.8)\n",

"Requirement already satisﬁed: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.7/dist- packages (from google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.10,>=2.9->tensorﬂow) (1.3.1)\n",

"Requirement already satisﬁed: importlib-metadata>=4.4 in /usr/local/lib/python3.7/dist- packages (from markdown>=2.6.8->tensorboard<2.10,>=2.9->tensorﬂow) (4.13.0)\n",

"Requirement already satisﬁed: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from importlib-metadata>=4.4->markdown>=2.6.8->tensorboard<2.10,>=2.9->tensorﬂow) (3.10.0)\n",

"Requirement already satisﬁed: pyasn1<0.5.0,>=0.4.6 in /usr/local/lib/python3.7/dist- packages (from pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorboard<2.10,>=2.9-

>tensorﬂow) (0.4.8)\n",

"Requirement already satisﬁed: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist- packages (from requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorﬂow) (3.0.4)\n",

"Requirement already satisﬁed: certiﬁ>=2017.4.17 in /usr/local/lib/python3.7/dist- packages (from requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorﬂow) (2022.9.24)\n",

"Requirement already satisﬁed: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in

/usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard<2.10,>=2.9-

>tensorﬂow) (1.24.3)\n",

"Requirement already satisﬁed: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorﬂow) (2.10)\n",

"Requirement already satisﬁed: oauthlib>=3.0.0 in /usr/local/lib/python3.7/dist-packages(from requests-oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.10,>=2.9-

>tensorﬂow) (3.2.2)\n",

"Requirement already satisﬁed: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/dist- packages (from packaging->tensorﬂow) (3.0.9)\n",

"Looking in indexes: https:/ pypi.org/simple, https:/ us-python.pkg.dev/colab- wheels/public/simple/\n",

"Requirement already satisﬁed: opencv-python in /usr/local/lib/python3.7/dist-packages (4.6.0.66)\n",

"Requirement already satisﬁed: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages(from opencv-python) (1.21.6)\n"

]

}

]

},

{

"cell\_type": "code", "source": [

"from keras.models import Sequential\n", "from keras.layers import Dense\n",

"from keras.layers import Convolution2D\n", "from keras.layers import MaxPooling2D\n", "from keras.layers import Flatten"

],

"metadata": {

"id": "LXQp5JUUy8by"

},

"execution\_count": 3,

"outputs": []

},

{

"cell\_type": "code", "source": [

"from tensorﬂow.keras.preprocessing.image import ImageDataGenerator\n", "train

= ImageDataGenerator(rescale=1/255)\n", "test = ImageDataGenerator(rescale=1/255)"

],

"metadata": {

"id": "-joU1JMNzBID"

},

"execution\_count": 4, "outputs": []

},

{

"cell\_type": "code", "source": [

"pwd"

],

"metadata": {

"colab": {

"base\_uri": "https:/ localhost:8080/", "height": 35

},

"id": "3zbBmApYzzob",

"outputId": "fa8fb36a-473c-4662-dbf7-67598141fa83"

},

"execution\_count": 8, "outputs": [

{

"output\_type": "execute\_result", "data": {

"text/plain":

[

"'/content'"

],

"application/vnd.google.colaboratory.intrinsic+json": { "type": "string"

}

},

"metadata": {}, "execution\_count": 8

}

]

},

{

"cell\_type": "code", "source": [

"import os\n",

"ﬁlenames = os.listdir('/content/drive/MyDrive/train\_set')"

],

"metadata": {

"id": "hNu0gAxNz5wV"

},

"execution\_count": 9, "outputs": []

},

{

"cell\_type": "code", "source": [

"x\_train = train\_dataset = train.ﬂow\_from\_directory(\"/content/drive/MyDrive/train\_set\",\n", " target\_size= (64,64),\n",

" batch\_size = 32,\n",

" class\_mode = 'binary')\n",

"x\_test = test\_dataset = test.ﬂow\_from\_directory(\"/content/drive/MyDrive/test\_set\",\n", " target\_size= (64,64),\n",

" batch\_size = 32,\n",

" class\_mode = 'binary')"

],

"metadata": {

"colab": {

"base\_uri": "https:/ localhost:8080/"

},

"id": "Ewd7ALq80I2h",

"outputId": "707c55af-f9b0-4164-e2cf-d642448ec7d1"

},

"execution\_count": 10, "outputs": [

{

"output\_type": "stream", "name": "stdout", "text": [

"Found 442 images belonging to 2 classes.\n", "Found 121 images belonging to 2 classes.\n"

]

}

]

},

{

"cell\_type": "code", "source": [ "x\_test.class\_indices"

],

"metadata": {

"colab": {

"base\_uri": "https:/ localhost:8080/"

},

"id": "Ypg9hbSD0VMb",

"outputId": "ddf6fee2-231b-4b2a-fc45-156d2c968517"

},

"execution\_count": 11, "outputs": [

{

"output\_type": "execute\_result", "data": {

"text/plain": [

"{'forest': 0, 'with ﬁre': 1}"

]

},

"metadata": {}, "execution\_count": 11

}

]

},

{

"cell\_type": "code", "source": [

"model = Sequential()"

],

"metadata": {

"id": "qyQ20wPg0XZg"

},

"execution\_count": 12,

"outputs": []

},

{

"cell\_type": "code", "source": [

"model.add(Convolution2D(32,(3,3),activation='relu',input\_shape=(64,64,3)))"

],

"metadata": {

"id": "3CCe1wJK0dq6"

},

"execution\_count": 13, "outputs": []

},

{

"cell\_type": "code", "source": [

"model.add(MaxPooling2D(2,2))"

],

"metadata": {

"id": "KiB6bTwt0gYl"

},

"execution\_count": 14, "outputs": []

},

{

"cell\_type": "code", "source": [ "model.add(Flatten())"

],

"metadata": {

"id": "gezPUvME0ixZ"

},

"execution\_count": 15, "outputs": []

},

{

"cell\_type": "code", "source": [

"model.add(Dense(512,activation='relu'))\n", "model.add(Dense(1,activation='sigmoid'))"

],

"metadata": {

"id": "jlTuRAuQ0l7j"

},

"execution\_count": 16, "outputs": []

},

{

"cell\_type": "code", "source": [

"model.compile(optimizer=\"adam\",loss=\"binary\_crossentropy\",metrics=[\"accuracy\"])"

],

"metadata": {

"id": "3VRs-oXD0quq"

},

"execution\_count": 17, "outputs": []

},

{

"cell\_type": "code", "source": [

"model.ﬁt(x\_train,steps\_per\_epoch=14

,epochs=10,validation\_data=x\_test,validation\_steps=4)"

],

"metadata": {

"colab": {

"base\_uri": "https:/ localhost:8080/"

},

"id": "7Ek-Gm6P0vpW",

"outputId": "f6a1472f-709a-4f36-9d4f-fa1946838e10"

},

"execution\_count": 18,

"outputs": [

{

"output\_type": "stream", "name": "stdout", "text": [

"Epoch 1/10\n",

"14/14 [==============================] - 157s 11s/step - loss: 3.7764 - accuracy:

0.5928 - val\_loss: 0.3833 - val\_accuracy: 0.8182\n", "Epoch 2/10\n",

"14/14 [==============================] - 22s 2s/step - loss: 1.0376 - accuracy:

0.6855 - val\_loss: 0.1756 - val\_accuracy: 0.9339\n", "Epoch 3/10\n",

"14/14 [==============================] - 21s 1s/step - loss: 0.2968 - accuracy:

0.8688 - val\_loss: 0.1248 - val\_accuracy: 0.9835\n", "Epoch 4/10\n",

"14/14 [==============================] - 21s 2s/step - loss: 0.2413 - accuracy:

0.9072 - val\_loss: 0.1233 - val\_accuracy: 0.9504\n", "Epoch 5/10\n",

"14/14 [==============================] - 21s 2s/step - loss: 0.1790 - accuracy:

0.9321 - val\_loss: 0.0887 - val\_accuracy: 0.9669\n", "Epoch 6/10\n",

"14/14 [==============================] - 21s 2s/step - loss: 0.1427 - accuracy:

0.9457 - val\_loss: 0.0762 - val\_accuracy: 0.9752\n",

"Epoch 7/10\n",

"14/14 [==============================] - 21s 2s/step - loss: 0.1059 - accuracy:

0.9706 - val\_loss: 0.0514 - val\_accuracy: 0.9917\n", "Epoch 8/10\n",

"14/14 [==============================] - 21s 1s/step - loss: 0.0835 - accuracy:

0.9774 - val\_loss: 0.0272 - val\_accuracy: 1.0000\n", "Epoch 9/10\n",

"14/14 [==============================] - 21s 2s/step - loss: 0.0657 - accuracy:

0.9774 - val\_loss: 0.0266 - val\_accuracy: 0.9917\n", "Epoch 10/10\n",

"14/14 [==============================] - 23s 2s/step - loss: 0.0465 - accuracy:

0.9819 - val\_loss: 0.0153 - val\_accuracy: 1.0000\n"

]

},

{

"output\_type": "execute\_result", "data": {

"text/plain": [

"<keras.callbacks.History at 0x7f2c0d8ec590>"

},

"metadata": {}, "execution\_count": 18

}

},

{

"cell\_type": "code", "source": [

"model.save(\"forest1.h5\")"

,

"metadata": {

"id": "wqMJztaF00Qh"

},

"execution\_count": 19, "outputs": []

},

{

"cell\_type": "code", "source": [

"!tar -zcvf image-classiﬁcation-model\_new.tgz forest1.h5"

],

"metadata": {

"colab": {

"base\_uri": "https:/ localhost:8080/"

},

"id": "3jah6H9-2Znl",

"outputId": "ed7217d8-6994-4f98-b136-98b1ff5f6b8b"

},

"execution\_count": 20, "outputs": [

{

"output\_type": "stream", "name": "stdout", "text":

"forest1.h5\n"

}

},

{

"cell\_type": "code", "source": [

"ls -1"

],

"metadata": {

"colab": {

"base\_uri": "https:/ localhost:8080/"

},

"id": "V9oOmNUW2cWk",

"outputId": "c706f1e1-1c8d-4b26-d8dd-a4bcdfe60e5c"

},

"execution\_count": 21, "outputs":

{

"output\_type": "stream", "name": "stdout", "text":

"\u001b[0m\u001b[01;34mdrive\u001b[0m/\n", "forest1.h5\n",

"image-classiﬁcation-model\_new.tgz\n", "\u001b[01;34msample\_data\u001b[0m/\n

}

},

{

"cell\_type": "code", "source": [

"!pip install watson-machine-learning-client --upgrade" "metadata": {

"colab": {

"base\_uri": "https:/ localhost:8080/", "height": 1000

},

"id": "nQj\_2bZ62ns3",

"outputId": "103e599b-947e-46f0-eb02-0dd7142a2130"

},

"execution\_count": 22,

"outputs": [

{

"output\_type": "stream", "name": "stdout", "text": [

"Looking in indexes: https:/ pypi.org/simple, https:/ us-python.pkg.dev/colab-

wheels/public/simple/\n",

"Collecting watson-machine-learning-client\n",

" Downloading watson\_machine\_learning\_client-1.0.391-py3-none-any.whl (538 kB)\n", "\u001b[K |████████████████████████████████| 538 kB 7.0 MB/s

\n","\u001b[?25hRequirement already satisﬁed: requests in /usr/local/lib/python3.7/dist- packages (from watson-machine-learning-client) (2.23.0)\n",

"Requirement already satisﬁed: tqdm in /usr/local/lib/python3.7/dist-packages (from watson-machine-learning-client) (4.64.1)\n",

"Requirement already satisﬁed: tabulate in /usr/local/lib/python3.7/dist-packages (from watson-machine-learning-client) (0.8.10)\n",

"Collecting lomond\n",

" Downloading lomond-0.3.3-py2.py3-none-any.whl (35 kB)\n",

"Requirement already satisﬁed: certiﬁ in /usr/local/lib/python3.7/dist-packages (from watson-machine-learning-client) (2022.9.24)\n",

"Requirement already satisﬁed: urllib3 in /usr/local/lib/python3.7/dist-packages (from watson-machine-learning-client) (1.24.3)\n",

"Collecting boto3\n",

" Downloading boto3-1.26.11-py3-none-any.whl (132 kB)\n",

"\u001b[K |████████████████████████████████| 132 kB 53.7 MB/s \n","\u001b[?25hRequirement already satisﬁed: pandas in

/usr/local/lib/python3.7/dist-

packages (from watson-machine-learning-client) (1.3.5)\n", "Collecting ibm-cos-sdk\n",

" Downloading ibm-cos-sdk-2.12.0.tar.gz (55 kB)\n",

"\u001b[K |████████████████████████████████| 55 kB 3.9 MB/s \n","\u001b[?25hCollecting jmespath<2.0.0,>=0.7.1\n",

" Downloading jmespath-1.0.1-py3-none-any.whl (20 kB)\n","Collecting s3transfer<0.7.0,>=0.6.0\n",

" Downloading s3transfer-0.6.0-py3-none-any.whl (79 kB)\n",

"\u001b[K |████████████████████████████████| 79 kB 9.5 MB/s \n","\u001b[?25hCollecting botocore<1.30.0,>=1.29.11\n",

" Downloading botocore-1.29.11-py3-none-any.whl (9.9 MB)\n",

"\u001b[K |████████████████████████████████| 9.9 MB 45.4 MB/s \n","\u001b[?25hRequirement already satisﬁed: python-dateutil<3.0.0,>=2.1 in

/usr/local/lib/python3.7/dist-packages (from botocore<1.30.0,>=1.29.11->boto3->watson- machine-learning-client) (2.8.2)\n",

"Collecting urllib3\n",

" Downloading urllib3-1.26.12-py2.py3-none-any.whl (140 kB)\n",

"\u001b[K |████████████████████████████████| 140 kB 42.1 MB/s \n","\u001b[?25hRequirement already satisﬁed: six>=1.5 in

/usr/local/lib/python3.7/dist-

packages (from python-dateutil<3.0.0,>=2.1->botocore<1.30.0,>=1.29.11->boto3->watson- machine-learning-client) (1.15.0)\n",

"Collecting ibm-cos-sdk-core==2.12.0\n",

" Downloading ibm-cos-sdk-core-2.12.0.tar.gz (956 kB)\n",

"\u001b[K |████████████████████████████████| 956 kB 51.7 MB/s \n","\u001b[?25hCollecting ibm-cos-sdk-s3transfer==2.12.0\n",

" Downloading ibm-cos-sdk-s3transfer-2.12.0.tar.gz (135 kB)\n",

"\u001b[K |████████████████████████████████| 135 kB 54.2 MB/s \n","\u001b[?25hCollecting jmespath<2.0.0,>=0.7.1\n",

" Downloading jmespath-0.10.0-py2.py3-none-any.whl (24 kB)\n", "Collecting requests\n",

" Downloading requests-2.28.1-py3-none-any.whl (62 kB)\n",

"\u001b[K |████████████████████████████████| 62 kB 1.6 MB/s \n","\u001b[?25hRequirement already satisﬁed: charset-normalizer<3,>=2 in

/usr/local/lib/python3.7/dist-packages (from requests->watson-machine-learning-client)(2.1.1)\n", "Requirement already satisﬁed: idna<4,>=2.5 in /usr/local/lib/python3.7/dist-packages (from

requests->watson-machine-learning-client) (2.10)\n",

"Requirement already satisﬁed: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (from pandas->watson-machine-learning-client) (2022.6)\n",

"Requirement already satisﬁed: numpy>=1.17.3 in /usr/local/lib/python3.7/dist-packages(from pandas->watson-machine-learning-client) (1.21.6)\n",

"Building wheels for collected packages: ibm-cos-sdk, ibm-cos-sdk-core, ibm-cos-sdk- s3transfer\n",

" Building wheel for ibm-cos-sdk (setup.py) ... \u001b[?25l\u001b[?25hdone\n", " Created wheel for ibm-cos-sdk: ﬁlename=ibm\_cos\_sdk-2.12.0-py3-none-any.whl

size=73931

sha256=841189e9104158317d85f159529014a3c34da1db4455cc140ecfd657ba3ed2ef\n", " Stored in directory:

/root/.cache/pip/wheels/ec/94/29/2b57327cf00664b6614304f7958abd29d77ea0e5bbece2ea57\n", " Building wheel for ibm-cos-sdk-core (setup.py) ... \u001b[?25l\u001b[?25hdone\n", " Created wheel for ibm-cos-sdk-core: ﬁlename=ibm\_cos\_sdk\_core-2.12.0-py3-none-

any.whl size=562962 sha256=6dd5fd11a6eb4cc566eefe7e82e573055238fbc5bdafc2604c164f8a6fa02255\n",

" Stored in directory:

/root/.cache/pip/wheels/64/56/fb/5cd6f4f40406c828a5289b95b2752a4d142a9afb359244ed8

d\n",

" Building wheel for ibm-cos-sdk-s3transfer (setup.py) ...

\u001b[?25l\u001b[?25hdone\n",

" Created wheel for ibm-cos-sdk-s3transfer: ﬁlename=ibm\_cos\_sdk\_s3transfer-2.12.0- py3-none-any.whl size=89778 sha256=3c9215c3ddaa7fc31a8c3783a78b5e3aa7a4cb9ea8d7dc1178e709c1ccb392a8\n",

" Stored in directory:

/root/.cache/pip/wheels/57/79/6a/ffe3370ed7ebc00604f9f76766e1e0348dcdcad2b2e32df9e1

\n",

"Successfully built ibm-cos-sdk ibm-cos-sdk-core ibm-cos-sdk-s3transfer\n", "Installing collected packages: urllib3, requests, jmespath, ibm-cos-sdk-core, botocore,

s3transfer, ibm-cos-sdk-s3transfer, lomond, ibm-cos-sdk, boto3, watson-machine-learning-client\n", " Attempting uninstall: urllib3\n",

" Found existing installation: urllib3 1.24.3\n", " Uninstalling urllib3-1.24.3:\n",

" Successfully uninstalled urllib3-1.24.3\n","

Attempting uninstall: requests\n",

" Found existing installation: requests 2.23.0\n", " Uninstalling requests-2.23.0:\n",

" Successfully uninstalled requests-2.23.0\n",

"Successfully installed boto3-1.26.11 botocore-1.29.11 ibm-cos-sdk-2.12.0 ibm-cos-sdk- core-2.12.0 ibm-cos-sdk-s3transfer-2.12.0 jmespath-0.10.0 lomond-0.3.3 requests-2.28.1 s3transfer-

* + 1. urllib3-1.26.12 watson-machine-learning-client-1.0.391\n"

]

},

{

"output\_type": "display\_data", "data": {

"application/vnd.colab-display-data+json": { "pip\_warning": {

"packages": [ "requests", "urllib3"

]

}

}

},

"metadata": {}

}

]

},

{

"cell\_type": "code", "source": [

"def guid\_from\_space\_name(client, space\_name):\n", " space = client.spaces.get\_details()\n",

" #print(space)\n",

" return(next(item for item in space['resources']if item['entity'][\"name\"] == space\_name)['metadata']['id'])"

],

"metadata": {

"id": "QSDKfvy\_3H8Q"

},

"execution\_count": 25, "outputs": []

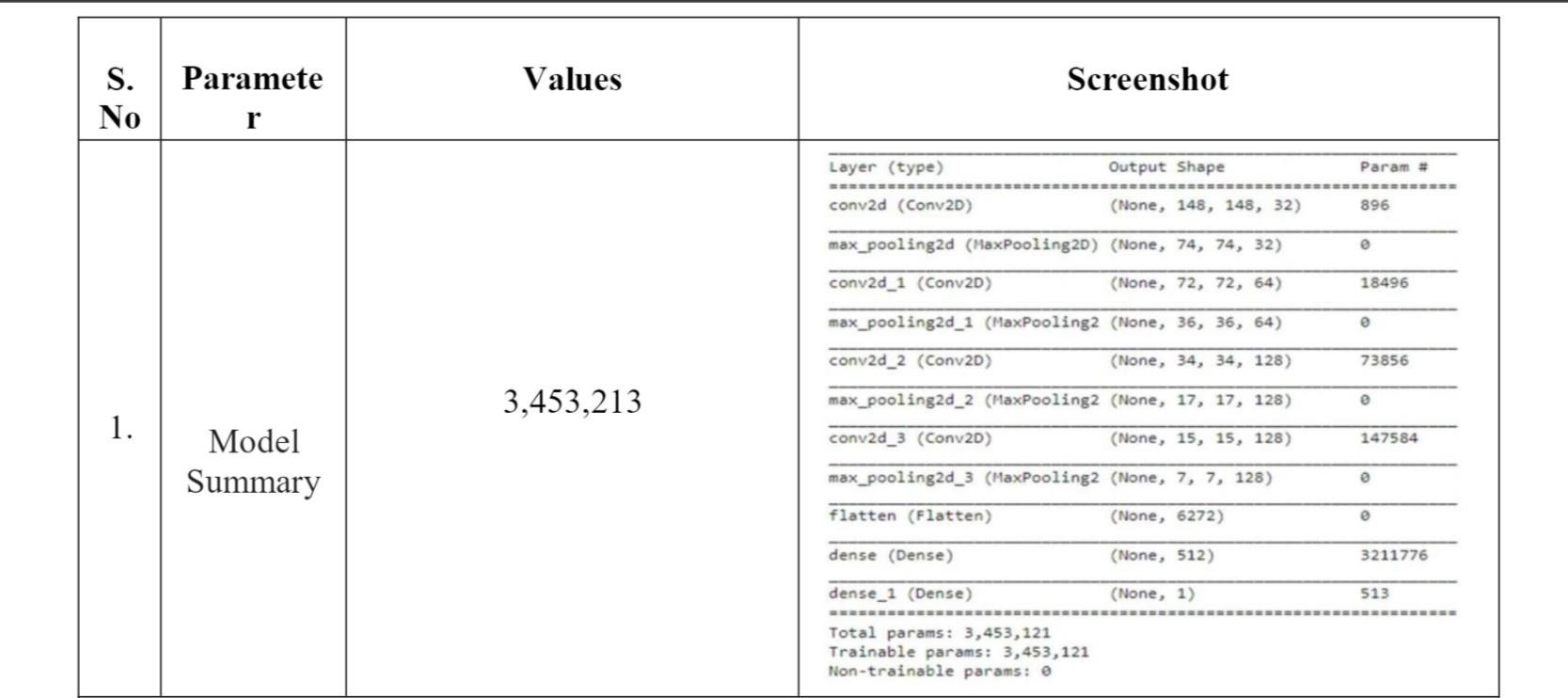
}

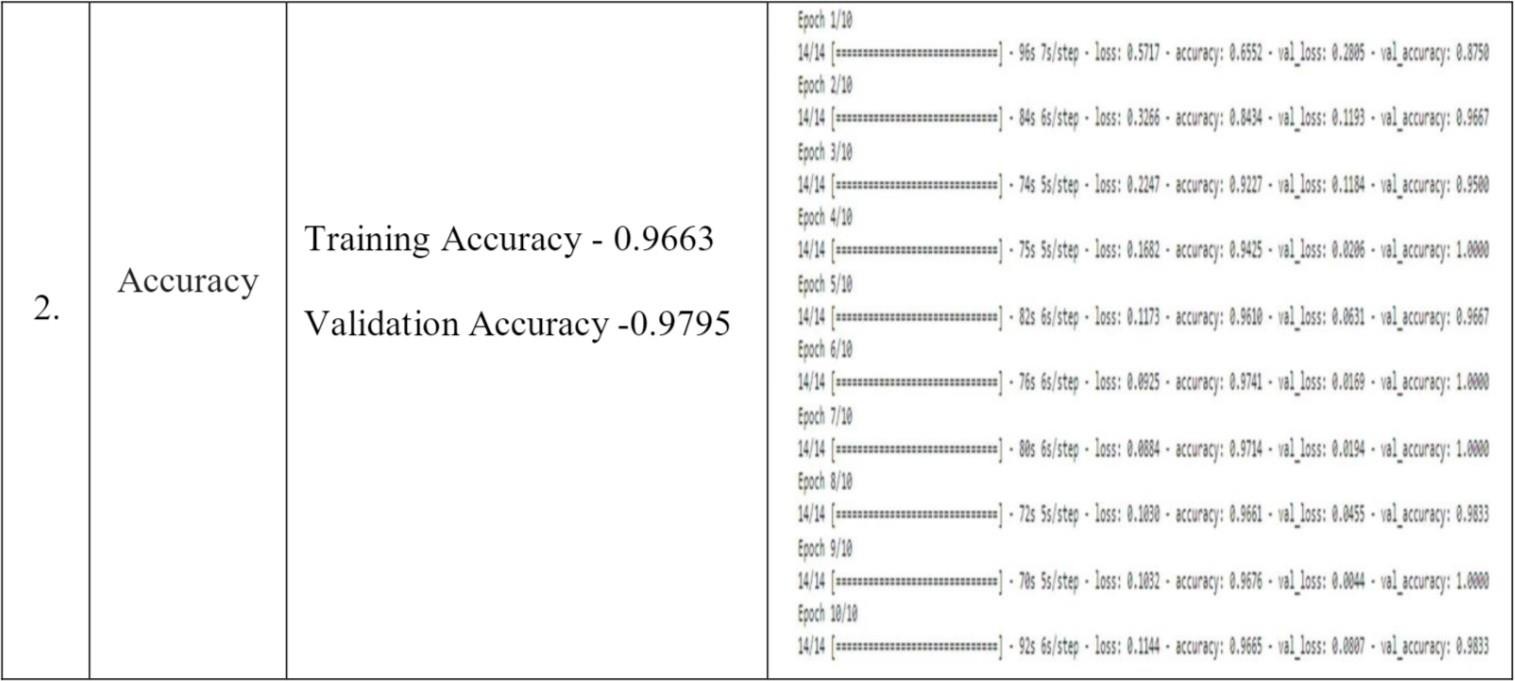
}

}

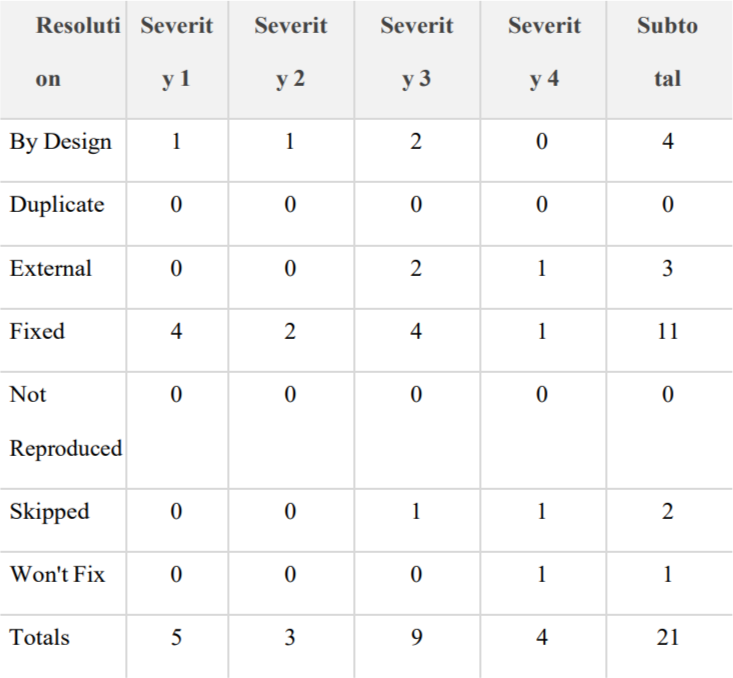
# TESTING AND RESULTS

* + - * 1. **Performance Testing**

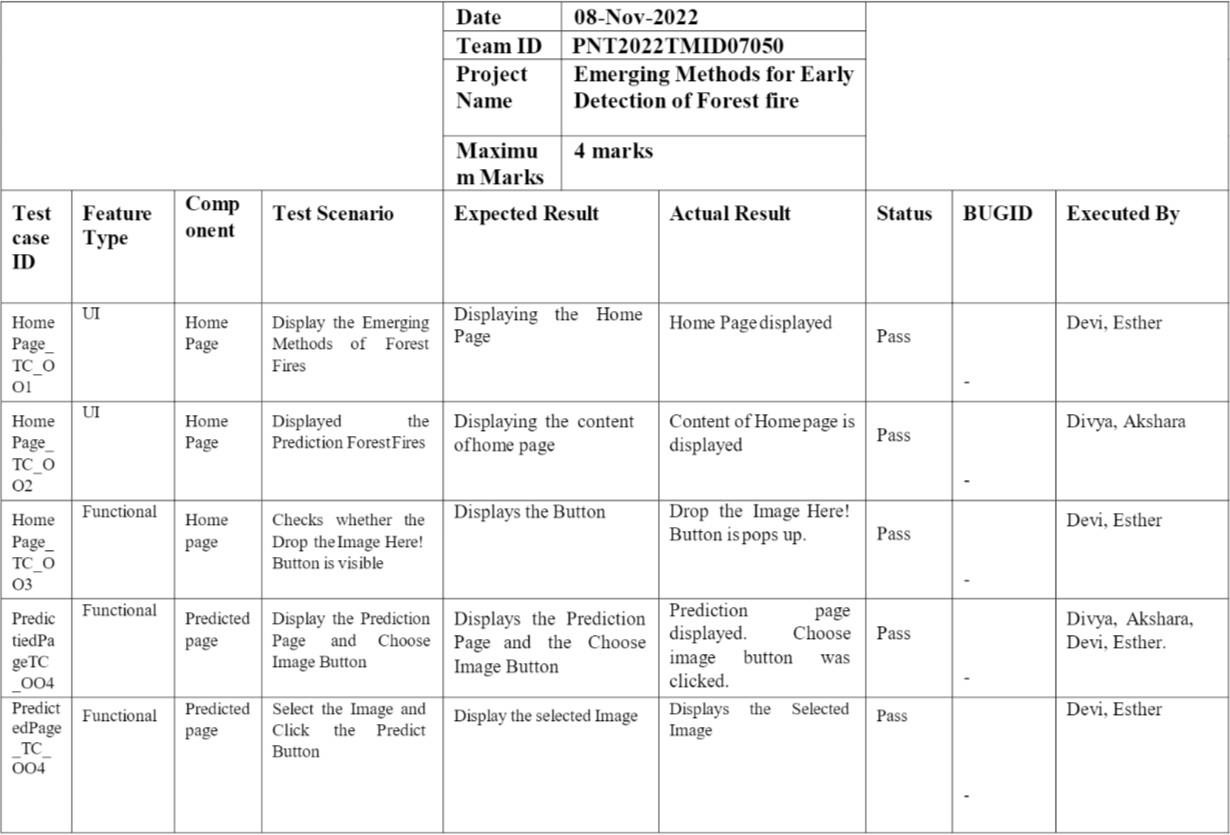




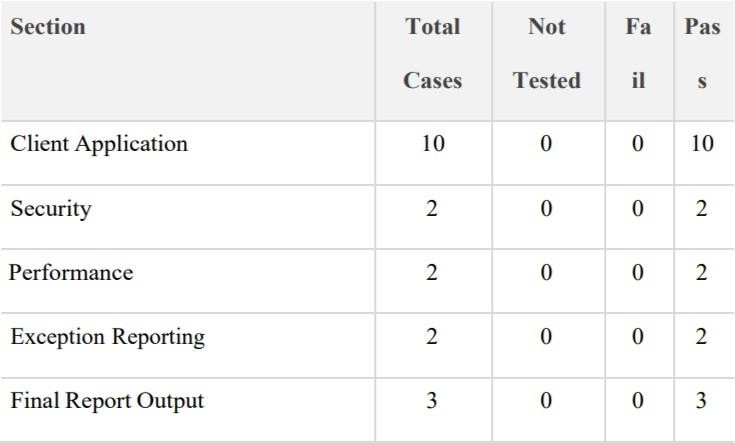
# User acceptance testing



1. **Test case**



# Test case analysis



1. **ENTIRE MODEL:**

*#Importing Keras libraries*

**import** keras

*#Importing ImageDataGenerator from Keras*

**from** matplotlib **import** pyplot **as** plt

**from** keras.preprocessing.image **import** ImageDataGenerator

*#Defining the Parameters*

train\_datagen**=**ImageDataGenerator(rescale**=**1.**/**255,shear\_range**=**0.2,rotation\_range**=**180,zoom

\_range**=**0.2,horizontal\_flip**=True**) test\_datagen**=**ImageDataGenerator(rescale**=**1.**/**255,shear\_range**=**0.2,rotation\_range**=**180,zoom\_ range**=**0.2,horizontal\_flip**=True**)

*#Applying ImageDataGenerator functionality to train dataset* x\_train**=**train\_datagen**.**flow\_from\_directory('/content/drive/MyDrive/train\_set',target\_size**=**(64, 64),batch\_size**=**32,class\_mode**=**'binary')

Found 442 images belonging to 2 classes.

x\_test**=**test\_datagen**.**flow\_from\_directory('/content/drive/MyDrive/test\_set',target\_size**=**(64,64), batch\_size**=**32,class\_mode**=**'binary')

Found 121 images belonging to 2 classes.

*#to define the linear Initialisation import sequential*

**from** keras.models **import** Sequential

*#to add layers import Dense*

**from** keras.layers **import** Dense

*#to create Convolutional kernel import convolution2D*

**from** keras.layers **import** Convolution2D

*#import Maxpooling layer*

**from** keras.layers **import** MaxPooling2D

*#import flatten layer*

**from** keras.layers **import** Flatten **import** warnings warnings**.**filterwarnings('ignore') *#Initializing the model*

model **=** Sequential()

*#Adding CNN Layers* model**.**add(Convolution2D(32,(3,3),input\_shape**=**(64,64,3),activation**=**'relu')) *#add maxpooling layers*

model**.**add(MaxPooling2D(pool\_size**=**(2,2)))

*#add faltten layer* model**.**add(Flatten()) *#Add Dense layers #add hidden layers*

model**.**add(Dense(150,activation**=**'relu')) *#add output layer* model**.**add(Dense(1,activation**=**'sigmoid')) *#configuring the learning process*

model**.**compile(loss**=**'binary\_crossentropy',optimizer**=**"adam",metrics**=**["accuracy"])

*#Training the model*

model**.**fit\_generator(x\_train,steps\_per\_epoch**=**14,epochs**=**10,validation\_data**=**x\_test,validation\_ steps**=**4)

Epoch 1/10

14/14 [==============================] - 152s 11s/step - loss: 1.2336 - accuracy: 0.6

244 - val\_loss: 0.3944 - val\_accuracy: 0.8760 Epoch 2/10

14/14 [==============================] - 23s 2s/step - loss: 0.3932 - accuracy: 0.839

4 - val\_loss: 0.1940 - val\_accuracy: 0.9421 Epoch 3/10

14/14 [==============================] - 22s 2s/step - loss: 0.2676 - accuracy: 0.891

4 - val\_loss: 0.1266 - val\_accuracy: 0.9835 Epoch 4/10

14/14 [==============================] - 23s 2s/step - loss: 0.2115 - accuracy: 0.907

2 - val\_loss: 0.0966 - val\_accuracy: 0.9587 Epoch 5/10

14/14 [==============================] - 22s 2s/step - loss: 0.1967 - accuracy: 0.923

1 - val\_loss: 0.0950 - val\_accuracy: 0.9752 Epoch 6/10

14/14 [==============================] - 24s 2s/step - loss: 0.1907 - accuracy: 0.925

3 - val\_loss: 0.1514 - val\_accuracy: 0.9256 Epoch 7/10

14/14 [==============================] - 23s 2s/step - loss: 0.2169 - accuracy: 0.907

1. - val\_loss: 0.0874 - val\_accuracy: 0.9669 Epoch 8/10

14/14 [==============================] - 26s 2s/step - loss: 0.1809 - accuracy: 0.925

1. - val\_loss: 0.0743 - val\_accuracy: 0.9669 Epoch 9/10

14/14 [==============================] - 23s 2s/step - loss: 0.1777 - accuracy: 0.929

9 - val\_loss: 0.0670 - val\_accuracy: 0.9917 Epoch 10/10

14/14 [==============================] - 26s 2s/step - loss: 0.2067 - accuracy: 0.909

5 - val\_loss: 0.0617 - val\_accuracy: 0.9917 *#Save the model* model**.**save("/content/drive/MyDrive/forest1.h5")

*#Predictions*

*#import load model from keras.model* **from** keras.models **import** load\_model *#import image from keras*

**from** tensorflow.keras.preprocessing **import** image

**import** numpy **as** np

*#import cv2*

**import** cv2

*#load the saved model* model**=**load\_model("/content/drive/MyDrive/forest1.h5") img**=**image**.**load\_img('/content/drive/MyDrive/test\_set/with fire/Forest\_fire\_MNRF\_esize\_IMG\_6743.jpg') x**=**image**.**img\_to\_array(img) res**=**cv2**.**resize(x,dsize**=**(64,64),interpolation**=**cv2**.**INTER\_CUBIC) *#expand the image shape*

x**=**np**.**expand\_dims(res,axis**=**0) pred**=**model**.**predict(x)

pred **=** int(pred[0][0]) pred

int(pred)

1/1 [==============================] - 0s 139ms/step

1

pip install twilio

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/sim ple/

Collecting twilio

Downloading twilio-7.15.3-py2.py3-none-any.whl (1.4 MB)

|████████████████████████████████| 1.4 MB 6.5 MB/s

Requirement already satisfied: pytz in /usr/local/lib/python3.7/dist-packages (from twilio) (202 2.6)

Requirement already satisfied: requests>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from t wilio) (2.23.0)

Collecting PyJWT<3.0.0,>=2.0.0

Downloading PyJWT-2.6.0-py3-none-any.whl (20 kB)

Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from req uests>=2.0.0->twilio) (2.10)

Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (fro m requests>=2.0.0->twilio) (2022.9.24)

Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (fro m requests>=2.0.0->twilio) (3.0.4)

Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python 3.7/dist-packages (from requests>=2.0.0->twilio) (1.24.3)

Installing collected packages: PyJWT, twilio Successfully installed PyJWT-2.6.0 twilio-7.15.3 **from** twilio.rest **import** Client

**if** pred**==**0:

print('Forest fire') account\_sid**=**'AC4c9a105651d0150d1b85af1bd4cf090c' auth\_token**=**'d18b90389f18b6069775b89c5c10ca1f' client**=**Client(account\_sid,auth\_token) message**=**client**.**messages \

**.**create(

body**=**'forest fire is detected,stay alert', *#use twilio free number* from\_**=**'+15134660214',

*#to number*

to**=**'+919361632961')

print(message**.**sid) print("Fire detected") print("SMS Sent!") **elif** pred**==**1:

print('No Fire')

No Fire

*#Open cv for video processing*

pip install twilio

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/sim ple/

Requirement already satisfied: twilio in /usr/local/lib/python3.7/dist-packages (7.15.3)

Requirement already satisfied: requests>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from t wilio) (2.23.0)

Requirement already satisfied: PyJWT<3.0.0,>=2.0.0 in /usr/local/lib/python3.7/dist-packages ( from twilio) (2.6.0)

Requirement already satisfied: pytz in /usr/local/lib/python3.7/dist-packages (from twilio) (202 2.6)

Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (fro m requests>=2.0.0->twilio) (3.0.4)

Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python 3.7/dist-packages (from requests>=2.0.0->twilio) (1.24.3)

Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (fro m requests>=2.0.0->twilio) (2022.9.24)

Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from req uests>=2.0.0->twilio) (2.10)

In [36]:

*#Creating An Account in Twilio Service #Sending Alert Message*

**from** logging **import** WARNING

*#import opencv library*

**import** cv2

*#import numpy*

**import** numpy **as** np

*#import image function from keras* **from** keras.preprocessing **import** image *#import load\_model from keras*

**from** keras.models **import** load\_model

*#import client from twilio API* **from** twilio.rest **import** Client *#import playsound package* **import** cv2

**import** numpy **as** np

**from** google.colab.patches **import** cv2\_imshow

**from** matplotlib **import** pyplot **as** plt

**import** librosa

**from** tensorflow.keras.preprocessing **import** image

**from** keras.models **import** load\_model

*# Create a VideoCapture object and read from input file*

*# If the input is the camera, pass 0 instead of the video file name*

cap **=** cv2**.**VideoCapture('/FOREST FIRE.mp4')

*# Check if camera opened successfully*

**if** (cap**.**isOpened()**== False**):

print("Error opening video stream or file")

*# Read until video is completed*

**while**(cap**.**isOpened()):

*# Capture frame-by-frame*

ret, frame **=** cap**.**read()

**if** ret **== True**:

cv2\_imshow(frame) x**=**image**.**img\_to\_array(frame)

res**=**cv2**.**resize(x,dsize**=**(64,64),interpolation**=**cv2**.**INTER\_CUBIC)

*#expand the image shape*

x**=**np**.**expand\_dims(res,axis**=**0) model**=**load\_model("/content/drive/MyDrive/forest1.h5") pred**=**model**.**predict(x)

pred **=** int(pred[0][0]) pred

int(pred)

**if** pred**==**0:

print('Forest fire')

# break else:

print("no danger")

# break

*# When everything done, release the video capture object*

cap**.**release()

*# Closes all the frames*

cv2**.**destroyAllWindows()



1/1 [==============================] - 0s 70ms/step

Forest fire

**from** twilio.rest **import** Client

**if** pred**==**0:

print('Forest fire')

**from** twilio.rest **import** Client account\_sid**=**'AC4c9a105651d0150d1b85af1bd4cf090c' auth\_token**=**'ee06c7d5053b02ef2ee7689157b255ee' client**=**Client(account\_sid,auth\_token) message**=**client**.**messages \

**.**create(

body**=**'forest fire is detected,stay alert', *#use twilio free number* from\_**=**'+15134660214',

*#to number*

to**=**'+919361632961')

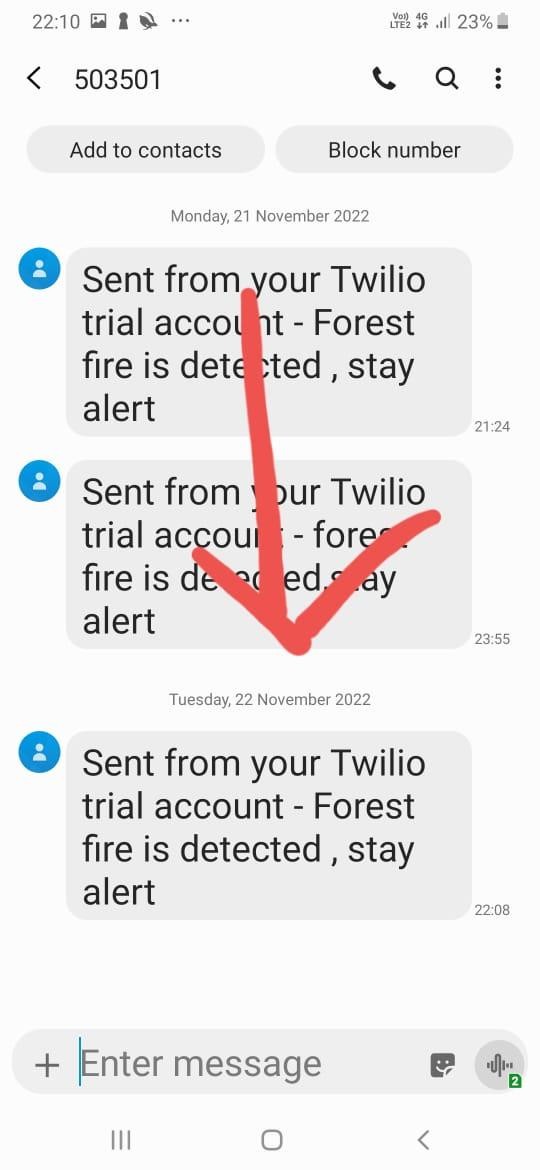
print(message**.**sid) print("Fire detected") print("SMS Sent!") **elif** pred**==**1:

print('No Fire') Forest fire

SM6c3521055b9c8a7899bfb240b5ea1b51 Fire detected

SMS Sent!

# OUTPUT SCREENSHOT:



1. **ADVANTAGES & DISADVANTAGES**

# ADVANTAGES:

* + The proposed system detects the forest ﬁre at a faster rate compared to existing system. It has enhanced data collection feature.
  + The major aspect is that it reduces false alarm and also has accuracy due to various sensors present.
  + It minimize the human effort as it works automatically. This is meagre -cost due to whichcan be easily accessed.
  + The main objective of our project is to receive an alert message through an app to the respective user.

# DISADVANTAGES:

* + - The electrical interference diminishes the potency of radio receiver.
    - The main drawback is that it has less coverage range areas

# CONCULSION

This type of system is the ﬁrst of its kind to ensure no further damage is then to forests when there is a ﬁre breakout and instantly a message is sent to the user through the App. Immediate response or early warning to a ﬁre breakout is mostly the only way to avoid losses and biology, cultural heritage damages to a great extent. Therefore the most important goals in ﬁre surveillance are quick and authentic detection of ﬁre. It is so much easier to suppress ﬁre while it is in its early stages. info about the progress of ﬁre is highly valuable for managing ﬁre during all its stages. Based on this data the ﬁreﬁghting staff can be guided on target to block ﬁre before it reaches cultural heritage sites and to suppress it quickly by utilizing required ﬁreﬁghting equipment and vehicles. With further research and invention, this project can be implemented in various forest areas so that we can save our forests and maintain great environs.

# FUTURE SCOPE

This project is far from complete and there is a lot of room for betterment. Some of the betterment that can be made to this project are as follows:

An Additional pump can be added so that it automatically sends water when there isa ﬁre breakout. Also industrial sensors can be used for better ranging and accuracy.

* This project has endless potential and can always be enhanced to become better.enforce this concept in the real world will beneﬁt several industries and reduce the workload on many workers, enhancing overall work eﬃciency.

# GitHub :

<https://github.com/IBM-EPBL/IBM-Project-19171-1659694051>